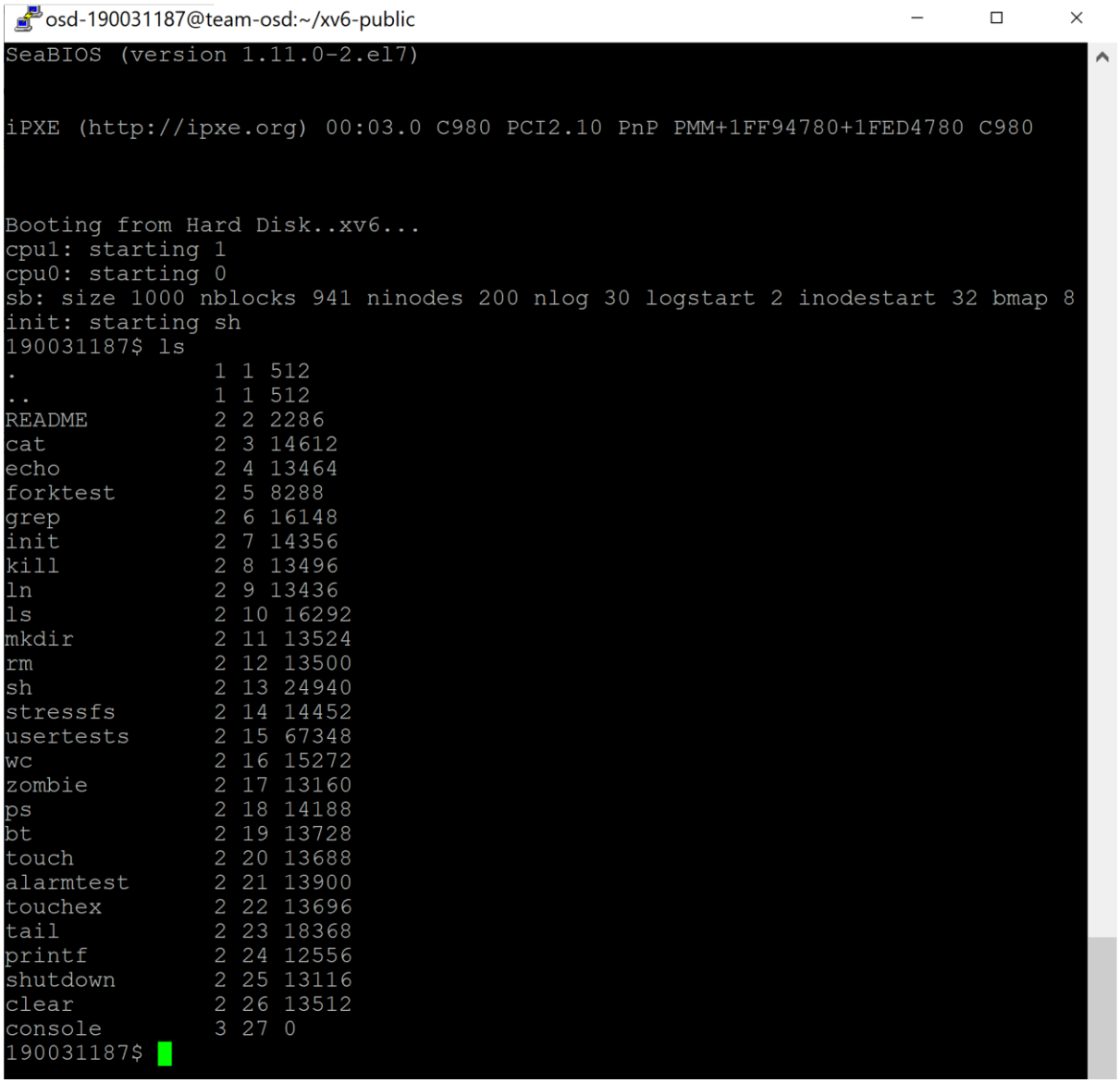
190031187 NERELLA VENKATA RADHAKRISHNA

Operating System and Design (19CS2106A)
Advanced Lab- 4

Xv6 design, implementation, and customization.

Clear:

```
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fs.h"
Int
main(int argc, char *argv[])
{
printf(1, "\033[2J\033[1;1H\n");
exit();
}
```

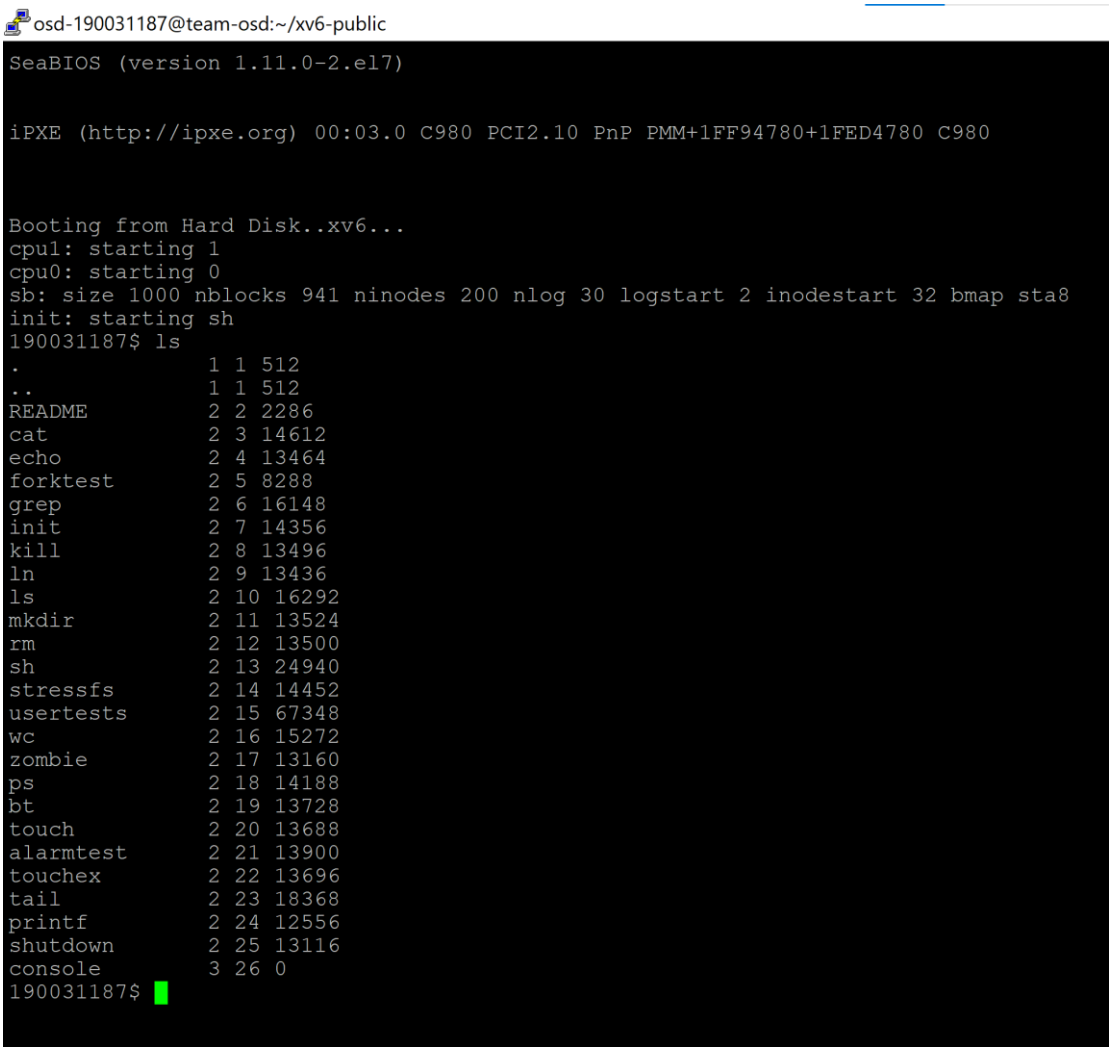


Shutdown

```
#include "types.h"
#include "stat.h"
#include "user.h"
int main(int argc, char *argv[])
{
halt();
exit();
}
```

we have to add halt system call to make it(shutdown) work
Add SYSCALL(halt) to usys.S
Add #define SYS_halt 22 to syscall.h
Add extern int sys_halt(void); and [SYS_halt] sys_halt, to syscall.c
Add int halt(void); to user.h
Add the following code to sysproc.c

```
int
sys_halt(void)
{
outb(0xf4, 0x00);
return 0;
}
```



UNIX system programming

1. **Link Library Shared libraries** (also called dynamic libraries) are linked into the program in two stages. First, during compile time, the link verifies that all the symbols (again, functions, variables and the like) required by the program, are either linked into the program, or in one of its shared libraries. To use a Library that is not linked into your program automatically by the compiler, you need to (1) include the library's header file in your C source file (test.c in the example below), and (2) tell the compiler to link in the code from the library .o file into your executable file

osd-190031187@team-osd:~/xv6-public

```
[osd-190031187@team-osd xv6-public]$ nano ctest1.c
[osd-190031187@team-osd xv6-public]$ nano ctest2.c
[osd-190031187@team-osd xv6-public]$ nano prog.c
[osd-190031187@team-osd xv6-public]$ ar -cvq libctest.a ctest1.o ctest2.o
a - ctest1.o
a - ctest2.o
[osd-190031187@team-osd xv6-public]$ ./prog
Valx=5
[osd-190031187@team-osd xv6-public]$
```

2. makefile

1. **Create file : nano hellomake.c**

```
#include <hellomake.h>

int main() {
  // call a function in another file
  myPrintHelloMake();

  return(0);
}
```

2. **Create file : nano hellofunc.c**

```
#include <stdio.h>
#include <hellomake.h>
void myPrintHelloMake(void) {
printf("Hello makefiles!\n");
return;
}
```

3. **Create file : nano hellomake.h**

```
/*
example include file
*/

void myPrintHelloMake(void);
```

To Compile the code execute the following command:

```
gcc -o hellomake hellomake.c hellofunc.c -I.
```

3. namedpipe: fifo

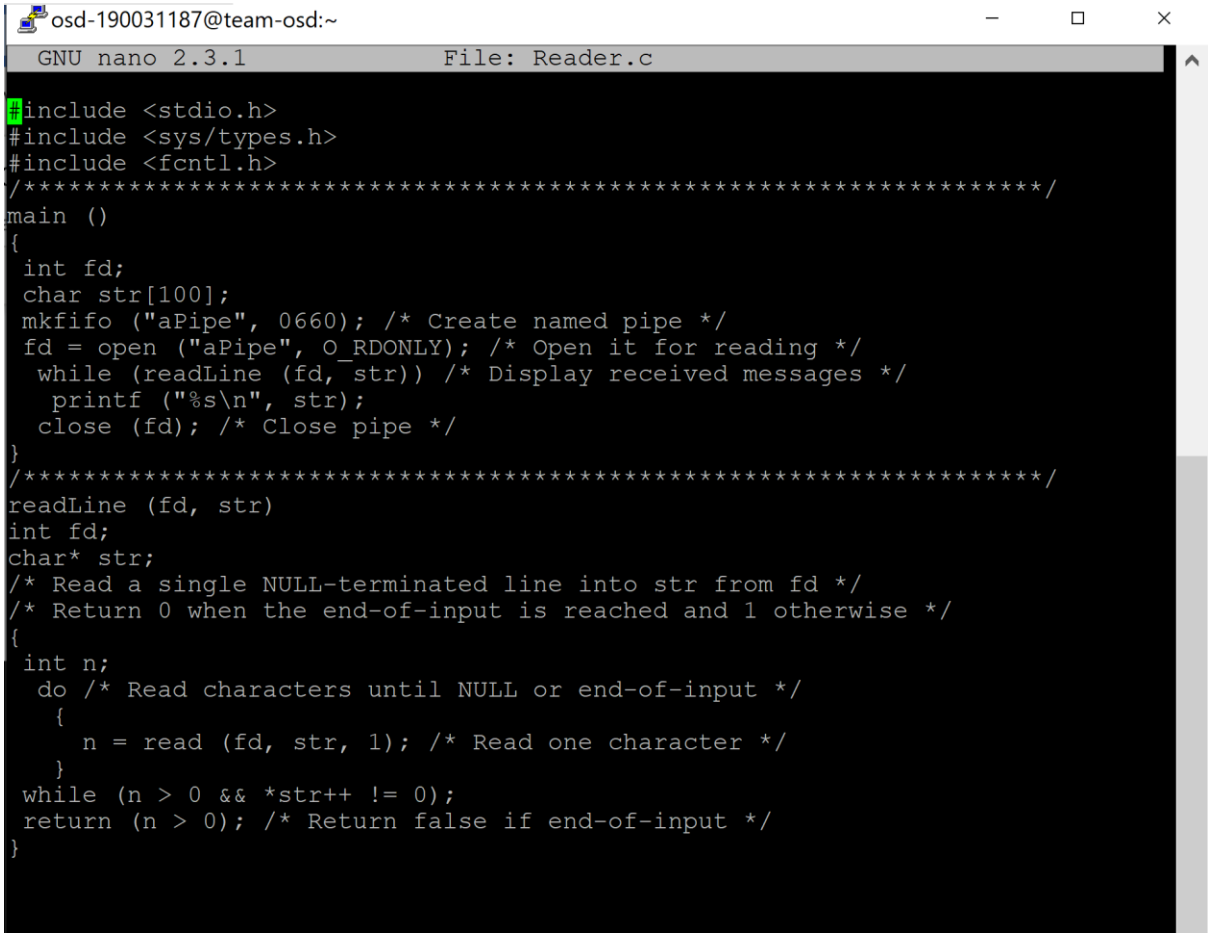
Reader.c

```
#include <stdio.h>
#include <sys/types.h>
#include <fcntl.h>
/*************************************************************************/
main ()
{
 int fd;
 char str[100];
 mkfifo ("aPipe", 0660); /* Create named pipe */
 fd = open ("aPipe", O_RDONLY); /* Open it for reading */
  while (readLine (fd, str)) /* Display received messages */
   printf ("%s\n", str);
  close (fd); /* Close pipe */
}
/*************************************************************************/
readLine (fd, str)
int fd;
char* str;
/* Read a single NULL-terminated line into str from fd */
/* Return 0 when the end-of-input is reached and 1 otherwise */
{
 int n;
  do /* Read characters until NULL or end-of-input */
   {
     n = read (fd, str, 1); /* Read one character */
   }
```

```
 while (n > 0 && *str++ != 0);
 return (n > 0); /* Return false if end-of-input */
}
```

Writer.c

```
#include <stdio.h>
#include <fcntl.h>
/*************************************************************************/
main ()
{
 int fd, messageLen, i;
 char message [100];
  /* Prepare message */
 sprintf (message, "Hello from PID %d", getpid ());
 messageLen = strlen (message) + 1;
  do /* Keep trying to open the file until successful */
    {
      fd = open ("aPipe", O_WRONLY); /* Open named pipe for writing */
      if (fd == -1) sleep (1); /* Try again in 1 second */
    }
 while (fd == -1);
  for (i = 1; i <= 3; i++) /* Send three messages */
    {
      write (fd, message, messageLen); /* Write message down pipe */
      sleep (3); /* Pause a while */
    }
  close (fd); /* Close pipe descriptor */
}
```



osd-190031187@team-osd:~

GNU nano 2.3.1 File: Writer.c

```
#include <stdio.h>
#include <fcntl.h>
#include <string.h>
/*****************************************************************/
main ()
{
 int fd, messageLen, i;
 char message [100];
  /* Prepare message */
 sprintf (message, "Hello from PID %d", getpid ());
 messageLen = strlen (message) + 1;
  do /* Keep trying to open the file until successful */
    {
      fd = open ("aPipe", O_WRONLY); /* Open named pipe for writing */
      if (fd == -1) sleep (1); /* Try again in 1 second */
    }
 while (fd == -1);
  for (i = 1; i <= 3; i++) /* Send three messages */
    {
      write (fd, message, messageLen); /* Write message down pipe */
      sleep (3); /* Pause a while */
    }
  close (fd); /* Close pipe descriptor */
}
```

osd-190031187@team-osd:~

```
[osd-190031187@team-osd ~]$ ./reader & ./reader & ./writer
[1] 58360
[2] 58361
Hello fro I 86
e o D56
mPD532HllofrmPI 832Hello from PID 58362
[1]-  Done                    ./reader
[2]+  Done                    ./reader
[osd-190031187@team-osd ~]$
```